

BeeSpace System Documentation: September 2007

| | |
|----------------------------------|----------|
| General Information ----- | 1 |
| Latest Version----- | 1 |
| Birds-Eye View----- | 1 |
| Utilizing Bigger Server----- | 2 |
| System Organization ----- | 3 |
| Datasets----- | 3 |
| Indexes----- | 3 |
| Databases----- | 3 |
| External Software----- | 3 |
| Internal Software----- | 3 |
| Common Library----- | 5 |

General Information

Latest Version

The most recent BeeSpace version (v3.5) is accessible via the URL:
http://beespace.igb.uiuc.edu:8080/BeeSpace3_v5/

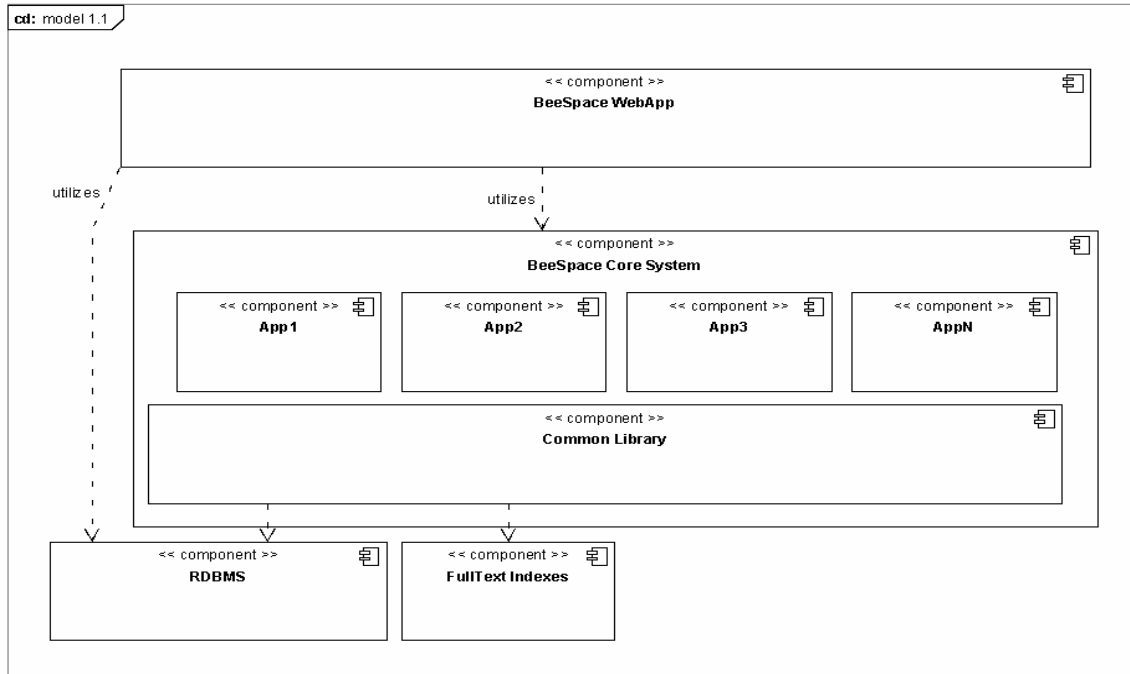
This recent changes in this version include:

- Improved Gene Annotator, allowing for meta-analysis of gene-by-phrase co-occurrence.
- Displaying breakout of term/phrase results when searching. For example, mouse-over query in search results page.
- Added Gene Find function. Does not do gene disambiguation currently.
- Sharing of spaces/regions between users. Regions/spaces can be set to private vs. public.
- Switching between users
- Field-based sorting of spaces/regions between users.
- Pre-generated region/space descriptions that are more meaningful.
- Confirm on delete and numerous other UI improvements.

Birds-Eye View

The following UML Component diagram captures the essence of the system from 10K feet. There exist two products (deliverables):

- a. **BeeSpace Web Application** – that allows end users to interact with our system via their browser, etc.
- b. **BeeSpace Core System** – encapsulating our core functionality and represents a potential deliverable in its own right (e.g. as a tar/zip file).



Utilizing Bigger Server

There is little effort needed to utilize a bigger server with more memory and CPUs. Most of this will be taken care transparently by the operating system. For example, more disk buffers will naturally be in memory. If two data-hogging processes are running concurrently, there's a much greater chance that neither will be forced to swap (for example, because the 128GB RAM is available as opposed to 32GB). For higher parallelization of the MI generation, the prun.sh script should be called with a larger "-P" argument (for example, "-P 15" for 16 CPUs).

What's Easy to change, hard to change, etc.

The easy to change items include web interface changes, using different application argument, and adding to or modifying the common library, etc. The medium effort changes include adding new functions to the system, adding or modifying DB tables, etc. Hard would probably be very low-level index changes simply because there are multiple programs that would need to be re-tested and evaluated for impact, etc. This latter item would require a fair amount of student coordination, discussions, etc. These are rather subjective evaluations (what's easy vs. what's hard), but this is my first reaction to this question.

System Organization

Datasets

The various datasets are located under /raid/data directory. These include the “Master” dataset which is Medline + Biosis + Books. Also, includes Biosis datasets and PMC (PubMed Central full text).

Additionally, some training datasets are also located under /raid/data/Training. Jing experimented with many of these.

Additionally, entities discovered by (Jing’s) NER programs are temporarily stored, for example, under /raid/data/Master/Entities before being loaded into DB.

Additionally, some user directories have been created under /raid/data in order to allow users to store and manipulate their own personalized datasets.

Indexes

The full-text indices are collocated with the datasets which they index. For example, the “Master” dataset has an Indri index located at /raid/data/Master/IndriSearch.

Databases

The MySQL database is used extensively for storage and retrieval of all data, excluding the full-text Indri/Lemur indices. Clearly some SQL and MySQL experience is assumed for working with these databases. SQL commands can be issued to show existing databases, tables and objects. In terms of database design, however, these are fairly simple databases with not many relations.

External Software

All 3rd party software components and applications that our system depends on are located under /raid/apps directory. This includes a diverse set of tools and libraries that our system utilizes. This is the best location to install new 3rd party software (COTS).

Internal Software

The root directory for all the BeeSpace software is /raid/soft. This includes sub-directories /BeeSpace/ (for v1), /BeeSpace2/ (for v2), and /BeeSpace3/ (for v3 and beyond). The breakout of subdirectories beyond this point is application-specific. Furthermore, it should be pointed out that BeeSpace2 and BeeSpace3 are close cousins of each other, while BeeSpace (v1) is significantly different.

The BeeSpace v3 functions include the following:

- **Gene Annotator** – provides text generated gene annotations by finding statistically significant terms/phrases in a foreground collection relative to a background collection. Located at /raid/soft/BeeSpace3/Annotator.
- **Gene Summarizer** – provides classification of relevant gene sentences according to fixed categories. Classification based on trained algorithm. Located at /raid/soft/BeeSpace3/GeneSum.
- **Search** – the main search functions are located under /raid/soft/BeeSpace3/Search. These provide the basic Boolean search capabilities and merging capabilities, etc, etc.
- **Theme Extraction** – the main theme extraction code and executables are located under /raid/soft/BeeSpace3/FlatTheme. This is the mixture modeling using EM algorithm. This includes the theme construction with priors, etc.
- **Small Worlds Clustering** – the binary files are located under /raid/soft/BeeSpace3/SWClusterer. However, I was never supplied the source code by student.
- **MI Generation** – generates an MI file with parallelization. Located under /raid/soft/BeeSpace3/Index/Apps. Run the prun.sh script with filter and index arguments. In order to accommodate more CPUs, the “-P” bandwidth parameter can be set when running this script.
- **Indexer** – various indexing and utility scripts exist under /raid/soft/BeeSpace3/Index in order to facilitate indexing. Stoplist files and other parameter files are also located under this directory.
- **Customized Indri/Lemur Toolkits** – the customized Lemur 4.4 and Indri 2.4 toolkits are located under /raid/soft/BeeSpace3/mod-lemur-4.4 and /raid/soft/BeeSpace3/mod-indri-2.4 respectively. This includes the modified tokenization strategy and other small changes.
- **Web Interface** – the web interface code lies under /raid/soft/BeeSpace3/Web directory and contains the entire j2ee web project.
- **Miscellaneous** – the /raid/soft/BeeSpace3/Utils contains various other programs and utilities that have been needed from time-to-time.

Common Library

The BeeSpace Common library encapsulates shared and reusable C++ code in the form of a compiled static library. The purpose of this library is to capture the common (generic) functionality, utility classes, etc., that the various applications can or do utilize. Ideally, as the Common Library grows in size, the numerous applications become smaller and quicker to develop, as well as more robust, etc. Currently, the Common Library encapsulates DB access code, Indri/Lemur extensions, metrics and function objects, and utility classes. In theory, it could grow to encapsulate some reusable statistical algorithms, data mining methods, etc, etc. The Common Library is located in its entirety at /raid/soft/BeeSpace3/Common.

Adding New Functionality

One can add new functionality to the Common Library by adding the requisite .cpp and .hpp files under the source directory. Then, modifying the Makefile to add the target object file obj/myfile.o to OBJS definition. Then, by copying and editing the target compile statement “\$(CC) ...” lines. Finally, one does a “make all” to guarantee that it builds, which internally also updates the static library.

Modifying Existing Functionality

One can modify existing Common Library functionality by editing the necessary source files and compiling (make all). To help locate hard-to-find bugs, one can always compile with -DDEBUG or -WALL flags as necessary or desired. Naturally, “gdb” debugger can also be used for debugging, etc.